FlatMatch: Bridging Labeled Data and Unlabeled Data with Cross-Sharpness for Semi-Supervised Learning

Zhuo Huang[1], **Li Shen**[2], **Jun Yu**[3], **Bo Han**[4], **Tongliang Liu**[1,*]

[1]Sydney AI Centre, The University of Sydney; [2]JD Explore Academy;
[3]Department of Automation, University of Science and Technology of China;
[4]Department of Computer Science, Hong Kong Baptist University
zhua0420@uni.sydney.edu.au, mathshenli@gmail.com, harryjun@ustc.edu.cn, bhanml@comp.hkbu.edu.hk, tongliang.liu@sydney.edu.au

Abstract

Semi-Supervised Learning (SSL) has been an effective way to leverage abundant unlabeled data with extremely scarce labeled data. However, most SSL methods are commonly based on instance-wise consistency between different data transformations. Therefore, the label guidance on labeled data is hard to be propagated to unlabeled data. Consequently, the learning process on labeled data is much faster than on unlabeled data which is likely to fall into a local minima that does not favor unlabeled data, leading to sub-optimal generalization performance. In this paper, we propose *FlatMatch* which minimizes a cross-sharpness measure to ensure consistent learning performance between the two datasets. Specifically, we increase the empirical risk on labeled data to obtain a worst-case model which is a failure case that needs to be enhanced. Then, by leveraging the richness of unlabeled data, we penalize the prediction difference (*i.e.*, cross-sharpness) between the worst-case model and the original model so that the learning direction is beneficial to generalization on unlabeled data. Therefore, we can calibrate the learning process without being limited to insufficient label information. As a result, the mismatched learning performance can be mitigated, further enabling the effective exploitation of unlabeled data and improving SSL performance. Through comprehensive validation, we show FlatMatch achieves state-of-the-art results in many SSL settings. Our code is available at https://github.com/tmllab/2023_NeurIPS_FlatMatch.

1 Introduction

Training deep models [31, 45] not only requires countless data but also hungers for label supervision which commonly consumes huge human laboring and unaffordable monetary costs. To ease the dependency on labeled data, semi-supervised learning (SSL) [6, 12, 27] has been one of the most effective strategies to exploit abundant unlabeled data by leveraging scarce labeled data. Traditional SSL commonly leverages unlabeled data by analyzing their manifold information. For example, semi-supervised support vector machine [6, 13] finds a classifier that crosses the low-density region based on large-margin theorem and label-propagation [60, 84] computes an affinity matrix in the input space to propagate the labels to their neighbor unlabeled data. Due to the computational burden of exploring manifold knowledge, recent advances in SSL benefit from sophisticated data augmentation techniques [4, 19, 20, 74, 80], they usually enforce predictive consistency between the original inputs and their augmented variants [75, 8, 7, 57, 66, 71, 76, 79], meanwhile using pseudo labels [46, 52, 53, 2] to guide the learning on unlabeled data.

[*]Corresponding to Tongliang Liu.

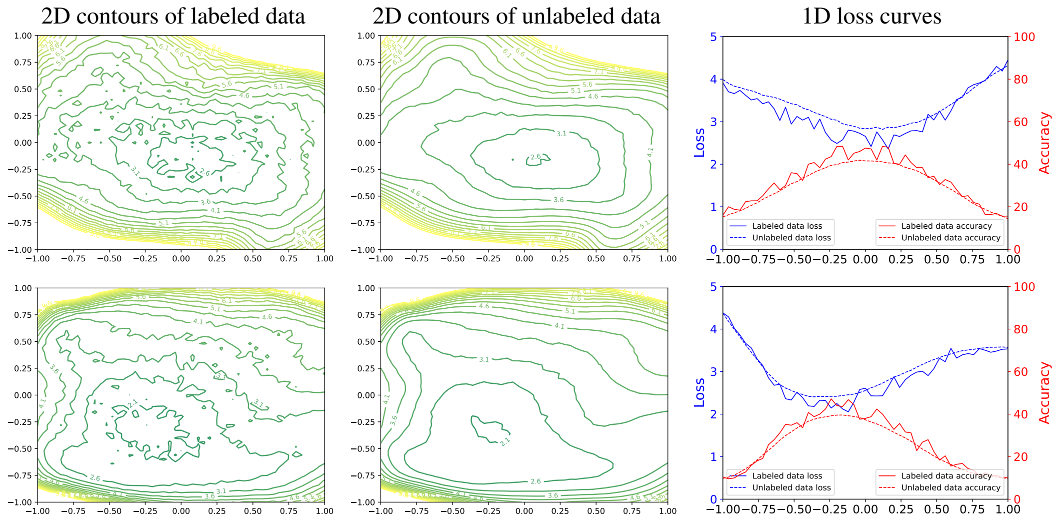

Figure 1: Loss landscapes of labeled data and unlabeled data obtained simultaneously from training using FixMatch [57] on CIFAR10 with 250 labels per class. The first row and second row show the results obtained from epoch 60 and epoch 150, respectively. The first column and second column show the 2D loss contours of labeled data and unlabeled data, respectively, and the last column shows the 1D loss curves.

However, as most of the existing methods only apply instance-wise consistency on each example with its own transformation, labeled data and unlabeled data are disconnected during training. Hence, the label guidance cannot be sufficiently propagated from the labeled set to the unlabeled set, causing the learning process on labeled data is much faster than on unlabeled data. As a result, the learning model could be misled to a local minima that cannot favor unlabeled data, leading to a non-negligible generalization mismatch. As generalization performance is closely related to the flatness of loss landscape [14, 24, 41, 51], we plot the loss landscapes of labeled data and unlabeled data using FixMatch [57] in Fig. 1[2]. We have two major findings: 1) The loss landscape of labeled data is extremely sharp, but the loss curve of unlabeled data is quite flat. 2) Moreover, the optimal loss value of labeled data is much lower than that of unlabeled data. Such two intriguing discoveries reveal the **essence** of SSL: *The learning on scarce labeled data convergences faster with lower errors than on unlabeled data, but it is vulnerable to perturbations and has an unstable loss curvature which rapidly increases as parameters slightly vary. Therefore, the abundant unlabeled data are leveraged so that SSL models are fitted to a wider space, thus producing a flatter loss landscape and generalizing better than labeled data*. Based on the analysis, existing methods that use instance-wise consistency with mismatched generalization performance could have two non-trivial pitfalls:

– The label supervision might not be sufficiently leveraged to guide the learning process of unlabeled data.
– The richness brought by unlabeled data remains to be fully exploited to enhance the generalization performance of labeled data.

As seeking the connection in input space is quite prohibitive, we resort to exploring the parameter space and propose *FlatMatch* which targets to encourage consistent learning performance between labeled data and unlabeled data, such that SSL models can obtain strong generalization ability without being limited by insufficient label information. Specifically, as the loss landscape on labeled data is quite unstable, we aim to mitigate this problem through applying an adversarial perturbation [24, 43, 69, 82, 83] to the model parameters so that the worst failure case can be found. Further, we enforce the worst-case model and the original model to achieve an agreement on unlabeled data via computing their prediction differences which are dubbed *cross-sharpness*. By minimizing such a cross-sharpness measure, the richness of unlabeled data can be fully utilized to calibrate the learning direction. In turn, the label information can be activated for producing accurate pseudo labels, thus successfully bridging the two datasets to enable improved SSL performance.

[2]Note that the loss curves cannot be directly generated using the raw data from training, because networks can fit all examples perfectly without showing any generalization evidence. Therefore, we load the data for plotting using slight data augmentation, *i.e.*, Random Rotation with 90°, which is very commonly used in SSL.

Furthermore, the proposed FlatMatch is easy to implement and can be adapted to complement many popular SSL methods [10, 15, 16, 28]. Although computing the proposed cross-sharpness normally requires an additional back-propagation step, we propose an efficient strategy so that our FlatMatch can be processed without extra computational burden. Through extensive experiments and analyses on various benchmark datasets, we show that our FlatMatch achieves huge performance improvement compared to some of the most powerful approaches. Specifically, on CIFAR100 dataset with 2500 labeled examples, our FlatMatch surpasses the foremost competitor so far with 1.11% improvement. In general, our contribution can be summarized into three points:

- We identify a generalization mismatch of SSL due to the disconnection between labeled data and unlabeled data, which leads to two critical flaws that remain to be solved (Section 1).
- We propose FlatMatch which addresses these problems by penalizing a novel cross-sharpness that helps improve the generalization performance of SSL (Section 4.2).
- We reduce the computational cost of FlatMatch by designing an efficient implementation (Section 4.3).
- Extensive experiments are conducted to fully validate the performance of FlatMatch, which achieves state-of-the-art results in several scenarios (Section 5).

2 Related Works

SSL has been developed rapidly since the booming trend of deep learning [31, 45, 25]. We roughly introduce it through three stages, namely traditional SSL, data-augmented SSL, and open-set SSL.

Traditional SSL: One of the first SSL methods is Pseudo Label [46] which has been very effective for training neural networks in a semi-supervised manner, which relies on the gradually improved learning performance to select confident samples [73, 72]. Another classical method is Co-training [9] which utilizes two separate networks to enforce consistent predictions on unlabeled data so that the classification can be more accurate. Inspired by co-training, many SSL approaches are motivated to conduct consistency regularization. Temporal ensembling [44] proposes to leverage historical model predictions to ensemble an accurate learning target that can be used to guide the learning of the current model. Mean Teacher [59] further suggests such an ensemble can be operated on the model weights to produce a teacher model that can provide improved learning targets. Instead of improving the teacher model, Noisy student [75] finds that adding noise to the student model can also improve SSL.

Data-Augmented SSL: Thanks to the development of data augmentation techniques and self-supervised learning [30, 34, 68, 67], further improvements in SSL are achieved by enforcing consistency between the original data and their augmented variants. VAT [49] proposes to add adversarial perturbations [25, 26] to unlabeled data. By using the pseudo-label guidance to leverage such adversarial unlabeled data, the model can be more robust to strong input noises, thus showing better performance on the test set. MixMatch [8] employs the MixUp [80] technique to interpolate between the randomly shuffled examples which largely smooths the input space as well as the label space, further enhancing the generalization result. Moreover, FixMatch [57] is based on strong augmentation such as AutoAugment [19] and RandAugment [20] to utilize the predictions of weakly augmented data to guide the learning of strongly augmented data, which can achieve near supervised learning performance on CIFAR10. Recently, Dash [76], FlexMatch [79], and FreeMatch [66] advanced SSL through investigating the threshold strategy for pseudo labels. They analyze the training dynamic of SSL and design various approaches to enhance the selection of unlabeled data, which effectively boosts the SSL performance to a new level.

Open-Set SSL: Moreover, another branch of SSL studies the realistic scenario where unlabeled data contains out-of-distribution (OOD) data [37, 38, 32, 33, 63, 64, 62]. The common goal is to alleviate the influence of OOD such that the learning process of SSL will not be misled. Several studies propose many techniques that can achieve this goal, such as distillation [17], meta-learning [29], large-margin regularization [10], content-style disentanglement [36], and consistency regularization can also help [56]. But in this paper, we mainly focus on the common assumption that labeled data and unlabeled are sampled independently and identically.

3 Preliminary: Improving Generalization via Penalizing Sharpness

Sharpness-Aware Minimization (SAM) [24, 43, 47, 69, 81] which focuses on optimizing the sharp points from parameter space so that the training model can produce a flat loss landscape. Specifically, given a set of fully-labeled data $\mathcal{D}^l = \{(x_i^l, y_i^l)\}_{i=1}^n$ containing n data points (x_i^l, y_i^l), SAM seeks to optimize a model parameterized by $\theta \in \Theta$ to fit the training dataset $\mathcal{D}^l$ so that θ can generalize well on a test set $\mathcal{D}^{te}$ drawn independently and identically as $\mathcal{D}^l$. Such an optimization process is normally conducted via minimizing the empirical risk $\mathcal{L}_l(\theta) = \frac{1}{n}\sum_{i=0}^n \ell_{ce}(\theta; g_\theta(x_i^l), y_i^l)$ which is realized by the cross-entropy loss ℓ_{ce} computed between the label prediction $f^l = g_\theta(x_i^l)$ and the class label y_i^l. To find the sharp points that maximally increase the empirical risk, SAM applies a weight perturbation $\epsilon \in \Theta$ to the model parameters θ and conducts the following inner maximization:

$$\epsilon^*(\theta) := \arg\max_{\|\epsilon\|_p \le \rho} \mathcal{L}_l(\theta + \epsilon) \approx \arg\max_{\|\epsilon\|_p \le \rho} \epsilon^\top \nabla_\theta \mathcal{L}_l(\theta) \overset{p=2}{\approx} \rho \frac{\nabla_\theta \mathcal{L}_l(\theta)}{\|\nabla_\theta \mathcal{L}_l(\theta)\|_2}, \tag{1}$$

where ϵ^* denotes the optimal perturbation which can be estimated using the gradient information of mini-batch inputs, ρ restricts the perturbation magnitude of ϵ within a ℓ_p-ball, and the approximation is given by the dual norm problem [24].

By perturbing the original model θ with ϵ^*, we can obtain the *sharpness* measure from SAM: $sharpness := \mathcal{L}_l(\theta + \epsilon^*) - \mathcal{L}_l(\theta)$ which indicates the how quickly loss changes under the worst-case model perturbation ϵ^*. To combine the sharpness term with empirical risk minimization (ERM), the SAM objective can be differentiated as follows:

$$\begin{aligned} \nabla_\theta \mathcal{L}_l^{SAM} :=& \nabla_\theta \left[\mathcal{L}_l(\theta + \epsilon^*(\theta)) - \mathcal{L}_l(\theta)\right] + \mathcal{L}_l(\theta) \approx \nabla_\theta \mathcal{L}_l(\theta + \epsilon^*(\theta)) \\ =& \frac{d(\theta + \epsilon^*(\theta))}{d\theta} \nabla_\theta \mathcal{L}_l(\theta)|_{\theta+\epsilon^*(\theta)} = \nabla_\theta \mathcal{L}_l(\theta)|_{\theta+\epsilon^*(\theta)} + o(\theta), \end{aligned} \tag{2}$$

where the last second-order term $o(\theta)$ can be discarded without significantly influencing the approximation. For detailed derivation and analysis, we refer to the original paper and related studies [1, 24, 82]. Intuitively, the gradient of SAM is computed on the worst-case point $\theta + \epsilon^*$, then such a gradient is utilized to update the original parameter θ. As a result, the model can produce a more flat loss landscape than ERM which would not change drastically, and being robust along the sharpest direction ϵ^* among its ℓ_p-norm neighbor.

Since the obtained flatness property has been demonstrated to have many benefits for generalization, such as being resistant to distribution shift [11, 39, 61], robustness against adversarial attack [69, 58], and effectiveness under label noise [24, 3, 35, 40, 70, 77], SAM has inspired many research progresses. However, SAM is only conducted under a fully-supervised setting, and whether or how can it be leveraged to improve learning with unlabeled data is still undiscovered. Next, we carefully introduce the proposed FlatMatch which improves the generalization performance of SSL by bridging labeled data and unlabeled data with a novel cross-sharpness.

4 FlatMatch: Semi-Supervised Learning with Cross-Sharpness

In this section, we carefully introduce our FlatMatch. First, we describe the SSL setting in a generalized way. Then, we demonstrate a novel regularization from FlatMatch which is dubbed cross-sharpness. Finally, we explain its optimization and design an efficient implementation.

4.1 General Semi-Supervised Learning

In SSL, we are commonly given a small set of labeled data $\mathcal{D}^l = \{(x_i^l, y_i^l)\}_{i=1}^n$ containing n labeled examples (x_i^l, y_i^l) and a large unlabeled data $\mathcal{D}^u = \{x_i^u\}_{i=1}^m$ containing m $(m \gg n)$ unlabeled examples x_i^u which are drawn independently and identically. Similar to previous notations, we aim to optimize a deep model $\theta \in \Theta$ in a semi-supervised manner so that θ can perform well on i.i.d sampled test set $\mathcal{D}^{te}$. The general objective for SSL is as follows:

$$\min_\theta \mathcal{L}_{ssl} = \min_\theta \mathcal{L}_l + \mathcal{L}_u = \min_\theta \frac{1}{n}\sum_{i=0}^n \ell_{ce}(\theta; g_\theta(x_i^l), y_i^l) + \frac{1}{m}\sum_{i=0}^m \mathbb{I}(\hat{y}_i > \tau)\ell_d(\theta; g_\theta(\mathcal{A}(x_i^u)), \hat{y}_i), \tag{3}$$

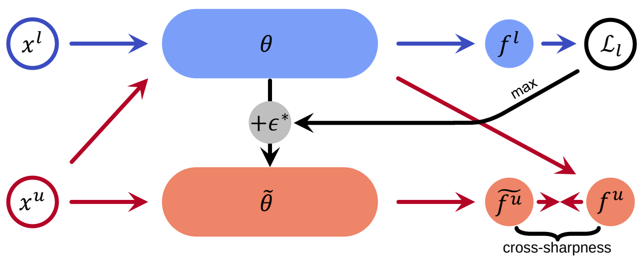

Figure 2: Illustration of FlatMatch. The blue arrows and red arrows denote the learning flows related to labeled data and unlabeled data, respectively; and the black arrows indicate the computation of the worst-case model.

where the first term is similar to Eq. 1 and denotes the empirical risk under the label supervision, the second term indicates the semi-supervised regularization for exploiting unlabeled data, the function $\ell_d(\cdot)$ denotes the loss criterion for unlabeled data, such as KL-divergence or cross-entropy loss, $\mathcal{A}(\cdot)$ stands for an augmentation function that transforms the original input x_i to its augmented variants, $\hat{y}_i$ represents the unsupervised learning target which can be a pseudo label or a guiding signal from a teacher model, and an indexing function $\mathbb{I}(\cdot)$ is commonly used to select the confident unlabeled data if their learning target surpasses a predefined threshold τ. Our FlatMatch replaces the second term with a cross-sharpness regularization as described below.

4.2 The Cross-Sharpness Regularization

The computation of cross-sharpness can be briefly illustrated in Fig. 2. Particularly, to solve the sharp loss landscape problem, as demonstrated in Section 1, we penalize the cross-sharpness regularization on each unlabeled example using a worst-case model derived from maximizing the loss on labeled data. Formally, such optimization is shown as follows:

$$\min_{\theta} \mathcal{R}_{X\text{-}sharp} := \min_{\theta} \frac{1}{m} \sum_{x^u \in \mathcal{D}^u} \ell_d(g_{\tilde{\theta}}(x^u), g_\theta(x^u)), \text{ where } \tilde{\theta} = \theta + \arg\max_{\|\epsilon\|_p \le \rho} \mathcal{L}_l, \tag{4}$$

in which $g_\theta(\cdot)$ indicates the forward propagation using the parameter θ whose prediction result is $f = g_\theta(\cdot)$, and $\tilde{\theta}$ stands for the worst-case model that maximally increases the empirical risk on labeled data. Here we drop the subscript i from each datum x for simplicity. By meaning "cross", the sharpness is not obtained by revisiting labeled data as SAM does, instead, we compute the sharpness by leveraging abundant unlabeled data to take full advantage of their generalization potential.

In detail, we leverage the original model θ to conduct a first propagation step as $f^l = g_\theta(x^l)$; $f^u = g_\theta(x^u)$, where f^l and f^u are label predictions from labeled data and unlabeled data, respectively. Then, the empirical risk on labeled data is computed as $\mathcal{L}_l = \sum_{x^l \in \mathcal{D}^l} \ell_{ce}(\theta; g_\theta(x^l), y^l)$. Following Eq. 1, we maximize the empirical risk $\mathcal{L}_l$ to obtain the weight perturbation as $\epsilon^* := \rho \frac{\nabla_\theta \mathcal{L}_l(\theta)}{\|\nabla_\theta \mathcal{L}_l(\theta)\|_2}$, further having the worst-case model $\tilde{\theta} = \theta + \epsilon^*$. Moreover, we conduct a second propagation step by passing the unlabeled data to $\tilde{\theta}$ and have $\tilde{f^u} = g_{\tilde{\theta}}(x^u)$. The second unlabeled data prediction $\tilde{f^u}$ is combined with the first prediction f^u obtained before to compute the cross-sharpness in Eq. 4[3].

Why does crossing the sharpness work? As shown by many theoretical signs of progress, the generalization error of SSL is largely dependent on the number of labels [5, 48]. Especially in recently proposed barely-supervised learning [57], the generalization error would be enormous. Although such a result may be pessimistic, unlabeled data can still be helpful. It is shown that when two distinct hypotheses on labeled data are co-regularized to achieve an agreement on unlabeled data, the generalization error can be reduced, as we quote: "The reduction is proportional to the difference between the representations of the labeled data in the two different views, where the measure of difference is determined by the unlabeled data" [54]. Intuitively, disconnecting labeled data and unlabeled data during training may be sub-optimal. To achieve better theoretical performance, the SSL model should focus on *learning something from unlabeled data that are* ***not contained*** *by labeled*

[3]Note that there is a slight difference between the sharpness from SAM and our cross-sharpness: The former one is measured by the change of cross-entropy loss under model perturbation. But in SSL, the pseudo labels of unlabeled data is not accurate enough compared to labeled data, so we use the difference between label predictions of θ and $\tilde{\theta}$ to measure our cross-sharpness.

Algorithm 1 FlatMatch and FlatMatch-e

```
Input: Labeled dataset $\mathcal{D}^l$, unlabeled dataset $\mathcal{D}^u$, model $\theta$, memory buffer $\mathcal{M}$ for storing historical gradients,
    and EMA factor $\alpha$ for updating buffer.
1: for $t \in 0, 1, \ldots, T-1$ do
2:     if Efficient training then                                                    ▷ FlatMatch-e
3:         Compute optimal perturbation as $\epsilon^* = \rho \frac{\mathcal{M}^t}{\|\mathcal{M}^t\|_2}$;
4:     else                                                                          ▷ FlatMatch
5:         Compute gradient as $\nabla_\theta \mathcal{L}_l(\theta) = \nabla_\theta \sum_{x^l \in \mathcal{D}^l} \ell_{ce}(\theta; g_\theta(x^l), y^l)$;   ▷ First propagation step
6:         Compute optimal perturbation as $\epsilon^* := \rho \frac{\nabla_\theta \mathcal{L}_l(\theta)}{\|\nabla_\theta \mathcal{L}_l(\theta)\|_2}$;
7:     Obtain worst-case model as $\tilde{\theta} = \theta + \epsilon^*$;
8:     Compute cross-sharpness via Eq. 4;
9:     Optimizing $\theta$ via Eq. 5, meanwhile save the gradient $\nabla_\theta \mathcal{L}_l(\theta)$;         ▷ Second propagation step
10:    Update memory buffer as $\mathcal{M}^{t+1} = (1-\alpha) \times \mathcal{M}^t + \alpha \times \nabla_\theta \mathcal{L}_l(\theta)$;
```

data. In our FlatMatch, we can achieve this goal by enforcing θ and $\tilde{\theta}$, which produces the maximal loss difference on labeled data, to be consistent in classifying unlabeled data. Therefore, FlatMatch can be more effective than other instance-wise consistent methods by bridging the knowledge of labeled data and unlabeled data for training.

4.3 Optimization and Efficient Implementation

To this end, we solve the optimization of FlatMatch by substituting $\mathcal{L}_u$ in Eq. 3 with $\mathcal{R}_{X\text{-}sharp}$ in Eq. 4:

$$\theta := \arg\min_\theta \mathcal{L}_l(\theta) + \mathcal{R}_{X\text{-}sharp}(\theta + \epsilon^*(\theta)) \approx \nabla_\theta \mathcal{L}_l + \nabla_\theta \mathcal{R}_{X\text{-}sharp}(\theta)\big|_{\theta + \rho \frac{\nabla_\theta \mathcal{L}_l(\theta)}{\|\nabla_\theta \mathcal{L}_l(\theta)\|_2}}. \quad (5)$$

Particularly, the second gradient is obtained on the worst-case point $\theta + \rho \frac{\nabla_\theta \mathcal{L}_l(\theta)}{\|\nabla_\theta \mathcal{L}_l(\theta)\|_2}$, then it is applied on the original model θ for gradient descent. As we can see, such optimization requires an additional back-propagation on labeled data, which doubles the computational complexity. Fortunately, our FlatMatch can be implemented efficiently with no extra computational burden than the baseline stochastic gradient descent (SGD) optimizer, which is dubbed FlatMatch-e. The implementation is summarized in Algorithm 1.

Although many efficient methods for implementing SAM have already been proposed [22, 23, 47, 83], they commonly need extra regularization or gradient decomposition, which are not harmonious to SSL and would complicate our method. In our scenario, FlatMatch has a critical difference from SAM regarding the two propagation steps: The gradients from the first step and second step are not computed on the same batch of data, *i.e.*, they are crossed from labeled data to unlabeled data. Since in SSL, each labeled batch and unlabeled batch are randomly coupled, we are allowed to use the gradient computed from the last batch of labeled data to obtain the cross-sharpness from the current batch of unlabeled data. Even better, we can use exponential moving average (EMA) [30, 44, 59] to stabilize the gradient so that our cross-sharpness can be computed accurately. Next, we conduct extensive experiments to carefully validate our approach.

5 Experiments

In this section, we conduct extensive comparisons and analyses to evaluate the proposed FlatMatch method. We first describe the experimental setup and implementation details. Then we compare FlatMatch with many edge-cutting SSL approaches to show the effectiveness of our method. Further, we conduct an ablation study to justify our design of FlatMatch. Finally, we demonstrate the efficiency, stability, and sensitivity of FlatMatch through various analytical studies.

5.1 Experimental Setup and Details

We follow the most common semi-supervised image classification protocols by using CIFAR10/100 [42] SVHN [50], and STL10 [18] datasets where a various number of labeled data are equally sampled from each class. Following Wang et al. [66], We choose Wide ResNet-28-2 [78] for CIFAR10 and SVHN, Wide ResNet-28-8 for CIFAR100, ResNet-37-2 [31] for STL10. All

Table 1: Error rates on CIFAR10/100, SVHN, and STL10 datasets. The fully-supervised results of STL10 are unavailable since we do not have label information for its unlabeled data. The best results are highlighted with **Bold** and the second-best results are highlighted with underline.

Dataset	CIFAR10				CIFAR100			SVHN			STL10	
# Label	10	40	250	4000	400	2500	10000	40	250	1000	40	1000
Π-Model [44]	79.18±1.11	74.34±1.76	46.24±1.29	13.13±0.59	86.96±0.80	58.80±0.66	36.65±0.00	67.48±0.95	13.30±1.12	7.16±0.11	74.31±0.85	32.78±0.40
Pseudo Label [46]	80.21±0.55	74.61±0.26	46.49±2.20	15.08±0.19	87.45±0.85	57.74±0.28	36.55±0.24	64.61±5.6	15.59±0.95	9.40±0.32	74.68±0.99	32.64±0.71
VAT [49]	79.81±1.17	74.66±2.12	41.03±1.79	10.51±0.12	85.20±1.40	46.84±0.79	32.14±0.19	74.75±3.38	4.33±0.12	4.11±0.20	74.74±0.38	37.95±1.12
Mean Teacher [59]	76.37±0.44	70.09±1.60	37.46±3.30	8.10±0.21	81.11±1.44	45.17±1.06	31.75±0.23	36.09±3.98	3.45±0.03	3.27±0.05	71.72±1.45	33.90±1.37
MixMatch [8]	65.76±7.06	36.19±6.48	13.63±0.59	6.66±0.26	67.59±0.66	39.76±0.48	27.78±0.29	30.60±8.39	4.56±0.32	3.69±0.37	54.93±0.96	21.70±0.68
ReMixMatch [7]	20.77±7.48	9.88±1.03	6.30±0.05	4.84±0.01	42.75±1.05	26.03±0.35	20.02±0.27	24.04±9.13	6.36±0.22	5.16±0.31	32.12±6.24	6.74±0.14
UDA [74]	34.53±10.69	10.62±3.75	5.16±0.06	4.29±0.07	46.39±1.59	27.73±0.21	22.49±0.23	5.12±4.27	1.92±0.05	1.89±0.01	37.42±8.44	6.64±0.17
FixMatch [57]	24.79±7.65	7.47±0.28	4.86±0.05	4.21±0.08	46.42±0.82	28.03±0.16	22.20±0.12	3.81±1.18	2.02±0.02	1.96±0.03	35.97±4.14	6.25±0.33
Dash [76]	27.28±14.09	8.93±3.11	5.16±0.23	4.36±0.11	44.82±0.96	27.15±0.22	21.88±0.07	2.19±0.18	2.04±0.02	1.97±0.01	34.52±4.30	6.39±0.56
MPL [52]	23.55±6.01	6.62±0.91	5.76±0.24	4.55±0.04	46.26±1.84	27.71±0.19	21.74±0.09	9.33±8.02	2.29±0.04	2.28±0.02	35.76±4.83	6.66±0.00
FlexMatch [79]	13.85±12.04	4.97±0.06	4.98±0.09	4.19±0.01	39.94±1.62	26.49±0.20	21.90±0.15	8.19±3.20	6.59±2.29	6.72±0.30	29.15±4.16	5.77±0.18
SoftMatch [16]	-	4.91±0.12	4.82±0.09	4.04±0.02	37.10±0.77	26.66±0.25	22.03±0.03	2.33±0.25	-	2.01±0.01	21.42±3.48	5.73±0.24
FreeMatch [66]	8.07±4.24	4.90±0.04	4.88±0.18	4.10±0.02	37.98±0.42	26.47±0.20	21.68±0.03	1.97±0.02	1.97±0.01	1.96±0.03	15.56±0.55	5.63±0.15
FlatMatch	15.23±8.67	5.58±2.36	**4.22**±1.14	**3.61**±0.49	38.76±1.62	**25.38**±0.85	**19.01**±0.43	2.46±0.06	**1.43**±0.05	**1.41**±0.04	16.20±4.34	**4.82**±1.21
FlatMatch-e	15.69±6.35	5.63±1.87	4.53±1.85	3.57±0.50	38.98±1.53	25.62±0.88	19.78±0.89	2.66±0.09	1.47±0.08	1.46±0.07	16.32±4.64	5.03±1.06
FlatMatch (Fix label)	**7.36**±5.62	**4.89**±1.24	**3.90**±1.72	**3.61**±0.49	**36.97**±0.95	**25.38**±0.85	**19.01**±0.43	**2.14**±0.05	**1.43**±0.05	**1.41**±0.04	**14.96**±0.67	**4.82**±1.21
Fully-Supervised	4.62±0.05				19.30±0.09			2.13±0.01			-	

SSL approaches are implemented using Pytorch framework, and the computation is based on 12GB NVIDIA 3090 GPU. We compare our method with well-known approaches, including Π-model [44], Pseudo Label [46], VAT [49], Mean Teacher [59], MixMatch [8], ReMixMatch [7], UDA [74], FixMatch [57], Dash [76], MPL [52], FlexMatch [79], SoftMatch [16], and FreeMatch [66].

To optimize all methods, we use SGD with a momentum of 0.9 with an initial learning rate of 0.03. We set the batch size as 64 for all datasets. Moreover, we set the weight decay value, pseudo label threshold τ, unlabeled batch ratio μ, and trade-off for SSL regularization as the same for all compared methods. To implement our FlatMatch, we set the perturbation magnitude $\rho = 0.1$, and EMA factor $\alpha = 0.999$ which is the same for Mean Teacher and other methods that require EMA. Moreover, in line 9 from Alg. 1, we need to separate the gradients regarding labeled data and unlabeled data in one back-propagation step. Such an operation can be done using "`functorch.make_functional_with_buffers`" in Pytorch, which can efficiently compute the gradients for each sample. For more details, please see the **appendix**.

5.2 Quantitative Comparison

The main comparison results are shown in Table 1, we can see that our FlatMatch achieves state-of-the-art results on many scenarios. For example, on CIFAR100 with 2500 labels, FlatMatch achieves 25.38% test errors, surpassing the second-best method, FreeMatch, for 1.09%; and on CIFAR100 with 10000 labels, FlatMatch achieves 19.01% performance, further improving the second-best result, ReMixMatch, for 1.01%, which even passes the fully-supervised baseline result for 0.29%. Moreover, FlatMatch reaches extremely low error rates in many settings. For instance, in CIFAR10 with 250 labels and 4000 labels, FlatMatch only has 4.22% and 3.61% error rates; and in SVHN with 250 labels and 1000 labels, FlatMatch can also surpass the fully-supervised learning by producing 1.43% and 1.41% error rates, respectively. Additionally, FlatMatch breaks the record on the STL10 dataset with 1000 labels by reaching 4.82% performance.

Moreover, we also show the performance of FlatMatch-e, an efficient variant of FlatMatch, which can also surpass many existing state-of-the-art methods. Such as in CIFAR100 with 10000 labels, FlatMatch-e can beat the second-best method, SoftMatch, with a 2.04% improvement. More importantly, FlatMatch-e only shows slight performance drops from FlatMatch but significantly reduces the computational burden, which is demonstrated in Section 5.4.

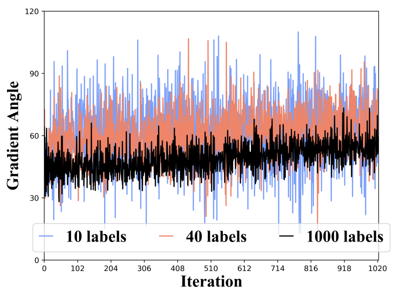

Figure 3: Gradient angle between cross-entropy loss $\mathcal{L}_l$ and general SSL loss $\mathcal{L}_{ssl}$.

Despite its effective performance, we also noticed that in some scenarios where the labels are extremely limited, such as CIFAR10 with 10 labels, CIFAR100 with 400 labels, etc, our FlatMatch is slightly worse than a recently proposed method FreeMatch [66]. This is because the gradient estimated by the extremely scarce labeled data cannot align with the general training direction. To intuitively demonstrate this issue, we show the gradient angle between the cross-entropy loss from labeled data and the total SSL loss in Fig. 3. We can see that the gradient angle oscillates stronger as

Algorithm 2 FlatMatch with fixed labels for Stabilizing Cross-Sharpness

```
Input: Labeled dataset $\mathcal{D}^l$, unlabeled dataset $\mathcal{D}^u$, model $\theta$, number of fixed labels $\#fix$, number of pre-train epochs $\#pre\_train$.
1: for $t \in 0, 1, \ldots, T-1$ do
2:     if $t \in 0, 1, \ldots, \#pre\_train$ then
3:         SSL pre-training by optimizing Eq. 3;
4:         if $t = \#pre\_train$ then
5:             Select Top-$\#fix$ confident unlabeled data $x_i^u$ and assign fixed labels $\hat{y}_i :=$ argmax$(g_\theta(x_i^u))$;
6:     else
7:         Using the augmented labeled data to compute cross-sharpness via Eq. 4;
8:         Using the original $\mathcal{D}^l$ and $\mathcal{D}^u$ to optimize $\theta$ via Eq. 5;
```

the number of labels gets smaller. When the label number reduces to only 10, the angle becomes extremely unstable and could surpass 90°. As a result, such a phenomenon could affect the worst-case perturbation computed by FlatMatch, which would introduce negligible noise to the training process.

However, this drawback can be properly solved by slightly augmenting the number of labels with pseudo-labeled unlabeled data. Specifically, before we start minimizing cross-sharpness, we first pre-train the backbone model with a common SSL method, such as FixMatch or FreeMatch, for 16 epochs to increase the confidence on unlabeled data. Then, we choose the most confident, *i.e.* high softmax probability, unlabeled data to be pseudo-labeled as labeled data. Different from other unlabeled data that are also trained with pseudo labels, the selected unlabeled data are fixed with their labels and act as labeled data when computing the sharpness. In this way, the cross-sharpness would be computed accurately by using both the original labeled data and augmented labeled data. Note that the augmentation with fixed labels is **only** used in the sharpness computation, when conducitng the second propagation step, all unlabeled data are treated similar to general SSL strategy, such process is summarized in Algorithm 2. Under this training strategy, we dub our method as "FlatMatch (Fix label)" and show its result in Table 1. The number of fixed labeled data in CIFAR10, CIFAR100, SVHM, and STL10 is set to 500, 2500, 500, and 1000, respectively. If the number of the original labeled dataset contains enough labels, we do not add more fixed labeled data. We can see that in this scenario, our method has the best performance in all settings including the ones with 1 or 4 labels in each class, which demonstrates that using more labels can largely stabilize sharpness computation and further benefits the performance of FlatMatch, thus achieving superior results on all settings.

5.3 Ablation Study

To carefully justify the design of our method, we compare FlatMatch with "sharp. on labeled data $\mathcal{D}^l$" and "sharp. on unlabeled data $\mathcal{D}^u$" where the former one denotes both the worst-case model $\hat{\theta}$ and sharpness is computed on labeled data and the latter one denotes $\hat{\theta}$ and sharpness are calculated on unlabeled data. Additionally, we compute the sharpness on the full dataset, as denoted by "sharp. on unlabeled data $\mathcal{D}^l \cup \mathcal{D}^u$". Moreover, we also analyze the effect of EMA smoothing on the performance of FlatMatch-e. Specifically, we compare FlatMatch-e with setting "w/o EMA" that just uses a gradient from the last batch of labeled data to calculate our worst-case model. The ablation study is conducted using CIFAR100 with a varied number of labels, which is shown in Table 2.

Table 2: Ablation study on CIFAR100.

Dataset	CIFAR100		
# Label	400	2500	10000
sharp. on $\mathcal{D}^l$	42.63±0.34	26.85±0.45	21.79±0.24
sharp. on $\mathcal{D}^u$	49.45±2.76	36.30±2.01	27.05±2.98
sharp. on $\mathcal{D}^l \cup \mathcal{D}^u$	43.88±1.64	27.26±1.62	23.42±1.77
FlatMatch	**38.76**±1.62	**25.38**±0.85	**19.01**±0.43
w/o EMA	40.64±0.97	29.44±1.56	23.23±1.28
FlatMatch-e	<u>38.98</u>±1.53	<u>25.62</u>±0.88	<u>19.78</u>±0.89

First, we can see that both two choices of our method are effective. Particularly, computing sharpness only on labeled data $\mathcal{D}^l$ shows smaller performance degradation than computing sharpness on unlabeled data $\mathcal{D}^u$, and it even shows better results than FixMatch. Hence, we know that the sharpness of labeled data can indeed improve the performance of SSL, only having limited performance because the number of labeled data is too scarce. On the other hand, when sharpness is completely based on unlabeled data, the performance significantly drops by nearly 10% compared to "sharpness on $\mathcal{D}^l$". This is because the training process on unlabeled data contains too much noise which causes erroneous gradient computation that would hinder the effectiveness of penalizing sharpness. Furthermore, we find that "w/o EMA" shows slightly inferior performance

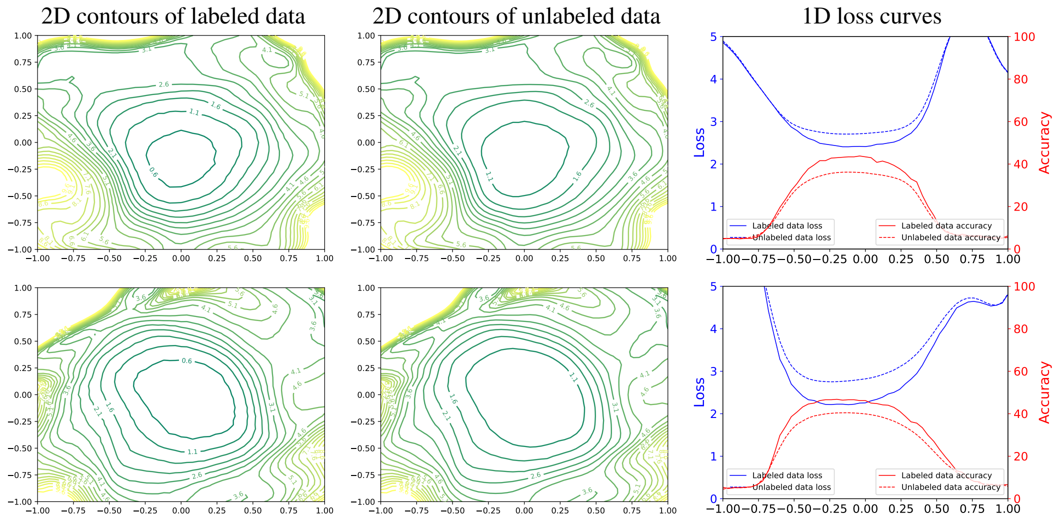

Figure 4: Loss landscapes of labeled data and unlabeled data obtained simultaneously from training using FlatMatch on CIFAR10 with 250 labels per class. The first row and second row show the results obtained from epoch 60 and epoch 150, respectively. The first column and second column show the 2D loss contours of labeled data and unlabeled data, respectively, and the last column shows the 1D loss curves.

to FlatMatch-e. Such degradation is consistent with the findings from Liu et al. [47] that the last gradient direction is distinct from the current one. As gradient descent has been conducted in the previous batch, reusing the gradient to perturb the current model might not find the perfect worst-case model on the current labeled data. Based on our observation, using EMA can stabilize the gradient can lead to accurate sharpness calculation.

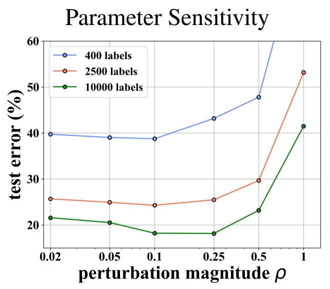

Figure 5: Parameter sensitivity analysis regarding perturbation magnitude ρ on CIFAR100.

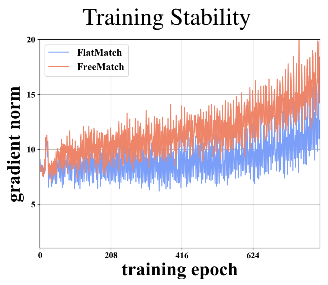

Figure 6: Stability analysis on gradient norm from training on CIFAR100.

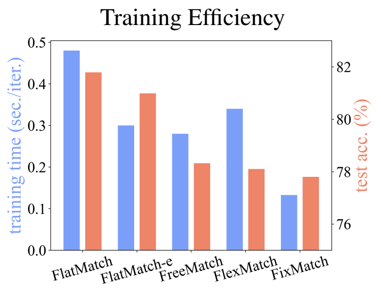

Figure 7: Training efficiency comparison on CIFAR100 with 10000 labels.

5.4 Analytical Study

In this section, we analyze the performance of FlatMatch by considering visualization, parameter sensitivity, training stability, and .efficiency.

Loss visualization: To show that FlatMatch can properly solve the sharp loss problem of labeled data, we train FlatMatch under the same setting as the ones demonstrated in Section 1 and plot the 2D loss contour as well as the 1D loss curve in Fig. 4. We can see that our FlatMatch can produce a much flatter loss landscape than FixMatch does in Fig. 1, where the jagged curve has been eliminated and become very smooth. Therefore, by minimizing cross-sharpness, the generalization performance on labeled data can be largely improved.

Sensitivity analysis: Our FlatMatch requires a hyperparameter ρ which controls the perturbation magnitude to obtain the worst-case model. To analyze the performance sensitivity of varying ρ, we show the result on CIFAR100 in Fig. 5. We observe that small ρ values show little impact on the test performance. However, when ρ increases to more than 0.25, the performance would largely degrade. Moreover, among three settings with varied label numbers, we find that more numbers labels could enhance the model sensitivity against changing of ρ. For example, when changing the ρ from 0.1

to 0.25, the performance difference on 400 labels, 2500 labels, and 10000 labels are 0.05%, 1.2%, 3.42%. Generally, the optimal value for ρ is 0.1.

Training stability: To validate the training stability of FlatMatch, we show the gradient norm which is an important criterion to present gradient variation in Fig. 6. We observe that the gradient norm of FlatMatch is significantly smaller than FreeMatch during the whole training phase. Therefore, minimizing the cross-sharpness can indeed improve training stability.

Training efficiency: We compare the training time and test accuracy of FlatMatch, FlatMatch-e, FreeMatch, FlexMatch, and FixMatch to validate the efficiency property of our method. As shown in Fig. 7, we find that despite the superior accuracy performance, FlatMatch requires more time for each iteration than the other three methods. However, the efficient variant FlatMatch-e can largely reduce the computational cost without losing too much learning performance. Hence, we can conclude that leveraging EMA for computing cross-sharpness is both effective and efficient.

6 Conclusion, Limitation and Broader Impact

In this paper, we propose a novel SSL method dubbed FlatMatch to address the mismatched generalization performance between labeled data and unlabeled data. By minimizing the proposed cross-sharpness regularization, FlatMatch can leverage the richness of unlabeled data to improve the generalization performance on labeled data. As a result, the supervised knowledge from labeled data can better guide SSL, and achieve improved test performance than most existing methods. Moreover, we propose an efficient implementation to reduce the computation cost. We conduct comprehensive experiments to validate our method regarding effectiveness, sensitivity, stability, and efficiency. Additionally, FlatMatch is slightly limited under barely-supervised learning due to the requirement of enough labeled data. Our method shows that SSL can be further improved by exploring generalization, which could be a potential direction in future research.

7 Acknowledgements

Li Shen was partially supported by STI 2030—Major Projects (2021ZD0201405). Zhuo Huang was supported by JD Technology Scholarship for Postgraduate Research in Artificial Intelligence (SC4103). Jun Yu was supported by the Natural Science Foundation of China (62276242), National Aviation Science Foundation (2022Z071078001), CAAI-Huawei MindSpore Open Fund (CAAIXSJLJJ-2021-016B, CAAIXSJLJJ-2022-001A), Anhui Province Key Research and Development Program (202104a05020007), USTC-IAT Application Sci. & Tech. Achievement Cultivation Program (JL06521001Y), and Sci. & Tech. Innovation Special Zone (20-163-14-LZ-001-004-01). Bo Han was supported by the NSFC Young Scientists Fund (62006202), NSFC General Program (62376235), and Guangdong Basic and Applied Basic Research Foundation (2022A1515011652). Tongliang Liu was partially supported by the following Australian Research Council projects: FT220100318, DP220102121, LP220100527, LP220200949, and IC190100031.

References

[1] Maksym Andriushchenko and Nicolas Flammarion. Towards understanding sharpness-aware minimization. In *ICML*, pages 639–668. PMLR, 2022.

[2] Yingbin Bai and Tongliang Liu. Me-momentum: Extracting hard confident examples from noisily labeled data. In *ICCV*, pages 9312–9321, 2021.

[3] Yingbin Bai, Erkun Yang, Bo Han, Yanhua Yang, Jiatong Li, Yinian Mao, Gang Niu, and Tongliang Liu. Understanding and improving early stopping for learning with noisy labels. In *NeurIPS*, volume 34, pages 24392–24403, 2021.

[4] Yingbin Bai, Erkun Yang, Zhaoqing Wang, Yuxuan Du, Bo Han, Cheng Deng, Dadong Wang, and Tongliang Liu. Rsa: Reducing semantic shift from aggressive augmentations for self-supervised learning. In *NeurIPS*, volume 35, pages 21128–21141, 2022.

[5] Shai Ben-David, Tyler Lu, and Dávid Pál. Does unlabeled data provably help? worst-case analysis of the sample complexity of semi-supervised learning. In *COLT*, pages 33–44, 2008.

[6] Kristin P Bennett and Ayhan Demiriz. Semi-supervised support vector machines. In *NeurIPS*, pages 368–374, 1999.

[7] David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*, 2019.

[8] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, pages 5049–5059, 2019.

[9] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, pages 92–100, 1998.

[10] Kaidi Cao, Maria Brbic, and Jure Leskovec. Open-world semi-supervised learning. In *ICLR*, 2022.

[11] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. In *NeurIPS*, volume 34, pages 22405–22418, 2021.

[12] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 20(3):542–542, 2009.

[13] Olivier Chapelle, Vikas Sindhwani, and Sathiya S Keerthi. Optimization techniques for semi-supervised support vector machines. *Journal of Machine Learning Research*, 9(2), 2008.

[14] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

[15] Baixu Chen, Junguang Jiang, Ximei Wang, Pengfei Wan, Jianmin Wang, and Mingsheng Long. Debiased self-training for semi-supervised learning. In *NeurIPS*, 2022.

[16] Hao Chen, Ran Tao, Yue Fan, Yidong Wang, Jindong Wang, Bernt Schiele, Xing Xie, Bhiksha Raj, and Marios Savvides. Softmatch: Addressing the quantity-quality trade-off in semi-supervised learning. In *ICLR*, 2023.

[17] Yanbei Chen, Xiatian Zhu, Wei Li, and Shaogang Gong. Semi-supervised learning under class distribution mismatch. In *AAAI*, 2020.

[18] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[19] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. In *CVPR*, 2018.

[20] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In *NeurIPS*, volume 33, pages 18613–18624, 2020.

[21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[22] Jiawei Du, Hanshu Yan, Jiashi Feng, Joey Tianyi Zhou, Liangli Zhen, Rick Siow Mong Goh, and Vincent YF Tan. Efficient sharpness-aware minimization for improved training of neural networks. In *ICLR*, 2022.

[23] Jiawei Du, Daquan Zhou, Jiashi Feng, Vincent Tan, and Joey Tianyi Zhou. Sharpness-aware training for free. In *NeurIPS*, volume 35, pages 23439–23451, 2022.

[24] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *ICLR*, 2020.

[25] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, pages 2672–2680, 2014.

[26] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

[27] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *NeurIPS*, pages 529–536, 2005.

[28] Lan-Zhe Guo and Yu-Feng Li. Class-imbalanced semi-supervised learning with adaptive thresholding. In *ICML*, pages 8082–8094. PMLR, 2022.

[29] Lan-Zhe Guo, Zhen-Yu Zhang, Yuan Jiang, Yu-Feng Li, and Zhi-Hua Zhou. Safe deep semi-supervised learning for unseen-class unlabeled data. In *ICML*, 2020.

[30] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[32] Rundong He, Zhongyi Han, Yang Yang, and Yilong Yin. Not all parameters should be treated equally: Deep safe semi-supervised learning under class distribution mismatch. In *AAAI*, pages 6874–6883. AAAI Press, 2022.

[33] Rundong He, Zhongyi Han, and Yilong Yin. Towards safe and robust weakly-supervised anomaly detection under subpopulation shift. *Knowledge-Based Systems*, 250:109088, 2022.

[34] Ziming Hong, Shiming Chen, Guo-Sen Xie, Wenhan Yang, Jian Zhao, Yuanjie Shao, Qinmu Peng, and Xinge You. Semantic compression embedding for generative zero-shot learning. In *IJCAI*, volume 7, pages 956–963, 2022.

[35] Huaxi Huang, Hui Kang, Sheng Liu, Olivier Salvado, Thierry Rakotoarivelo, Dadong Wang, and Tongliang Liu. Paddles: Phase-amplitude spectrum disentangled early stopping for learning with noisy labels. In *ICCV*, pages 16719–16730, 2023.

[36] Zhuo Huang, Xiaobo Xia, Li Shen, Bo Han, Mingming Gong, Chen Gong, and Tongliang Liu. Harnessing out-of-distribution examples via augmenting content and style. In *ICLR*, 2023.

[37] Zhuo Huang, Chao Xue, Bo Han, Jian Yang, and Chen Gong. Universal semi-supervised learning. In *NeurIPS*, volume 34, 2021.

[38] Zhuo Huang, Jian Yang, and Chen Gong. They are not completely useless: Towards recycling transferable unlabeled data for class-mismatched semi-supervised learning. *IEEE Transactions on Multimedia*, 2022.

[39] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[40] Hui Kang, Sheng Liu, Huaxi Huang, Jun Yu, Bo Han, Dadong Wang, and Tongliang Liu. Unleashing the potential of regularization strategies in learning with noisy labels. *arXiv preprint arXiv:2307.05025*, 2023.

[41] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *ICLR*, 2017.

[42] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[43] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *ICML*, pages 5905–5914. PMLR, 2021.

[44] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2016.

[45] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[46] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop*, 2013.

[47] Yong Liu, Siqi Mai, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Towards efficient and scalable sharpness-aware minimization. In *CVPR*, pages 12360–12370, 2022.

[48] Yury Maximov, Massih-Reza Amini, and Zaid Harchaoui. Rademacher complexity bounds for a penalized multi-class semi-supervised algorithm. *Journal of Artificial Intelligence Research*, 61:761–786, 2018.

[49] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 41(8):1979–1993, 2018.

[50] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop*, 2011.

[51] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *NeurIPS*, volume 30, 2017.

[52] Hieu Pham, Zihang Dai, Qizhe Xie, Minh-Thang Luong, and Quoc V Le. Meta pseudo labels. *arXiv preprint arXiv:2003.10580*, 2020.

[53] Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *ICLR*, 2021.

[54] David S Rosenberg and Peter L Bartlett. The rademacher complexity of co-regularized kernel classes. In *Artificial Intelligence and Statistics*, pages 396–403. PMLR, 2007.

[55] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[56] Kuniaki Saito, Donghyun Kim, and Kate Saenko. Openmatch: Open-set semi-supervised learning with open-set consistency regularization. In *NeurIPS*, volume 34, 2021.

[57] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020.

[58] David Stutz, Matthias Hein, and Bernt Schiele. Relating adversarially robust generalization to flat minima. In *ICCV*, pages 7807–7817, 2021.

[59] Antti Tarvainen and Harri Valpola. Meanteachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, pages 1195–1204, 2017.

[60] Fei Wang and Changshui Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 20(1):55–67, 2007.

[61] Pengfei Wang, Zhaoxiang Zhang, Zhen Lei, and Lei Zhang. Sharpness-aware gradient matching for domain generalization. *arXiv preprint arXiv:2303.10353*, 2023.

[62] Qizhou Wang, Zhen Fang, Yonggang Zhang, Feng Liu, Yixuan Li, and Bo Han. Learning to augment distributions for out-of-distribution detection. In *NeurIPS*, 2023.

[63] Qizhou Wang, Feng Liu, Yonggang Zhang, Jing Zhang, Chen Gong, Tongliang Liu, and Bo Han. Watermarking for out-of-distribution detection. In *NeurIPS*, 2022.

[64] Qizhou Wang, Junjie Ye, Feng Liu, Quanyu Dai, Marcus Kalander, Tongliang Liu, Jianye Hao, and Bo Han. Out-of-distribution detection with implicit outlier transformation. In *ICLR*, 2023.

[65] Yidong Wang, Hao Chen, Yue Fan, Wang Sun, Ran Tao, Wenxin Hou, Renjie Wang, Linyi Yang, Zhi Zhou, Lan-Zhe Guo, et al. Usb: A unified semi-supervised learning benchmark for classification. In *NeurIPS*, volume 35, pages 3938–3961, 2022.

[66] Yidong Wang, Hao Chen, Qiang Heng, Wenxin Hou, Marios Savvides, Takahiro Shinozaki, Bhiksha Raj, Zhen Wu, and Jindong Wang. Freematch: Self-adaptive thresholding for semi-supervised learning. In *ICLR*, 2023.

[67] Zhaoqing Wang, Ziyu Chen, Yaqian Li, Yandong Guo, Jun Yu, Mingming Gong, and Tongliang Liu. Mosaic representation learning for self-supervised visual pre-training. In *ICLR*, 2022.

[68] Zhaoqing Wang, Qiang Li, Guoxin Zhang, Pengfei Wan, Wen Zheng, Nannan Wang, Mingming Gong, and Tongliang Liu. Exploring set similarity for dense self-supervised representation learning. In *CVPR*, pages 16590–16599, 2022.

[69] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *NeurIPS*, volume 33, pages 2958–2969, 2020.

[70] Yuhao Wu, Xiaobo Xia, Jun Yu, Bo Han, Gang Niu, Masashi Sugiyama, and Tongliang Liu. Making binary classification from multiple unlabeled datasets almost free of supervision. *arXiv preprint arXiv:2306.07036*, 2023.

[71] Zhengning Wu, Tianyu He, Xiaobo Xia, Jun Yu, Xu Shen, and Tongliang Liu. Conditional consistency regularization for semi-supervised multi-label image classification. *IEEE Transactions on Multimedia*, 2023.

[72] Xiaobo Xia, Bo Han, Yibing Zhan, Jun Yu, Mingming Gong, Chen Gong, and Tongliang Liu. Combating noisy labels with sample selection by mining high-discrepancy examples. In *ICCV*, pages 1833–1843, 2023.

[73] Xiaobo Xia, Tongliang Liu, Bo Han, Mingming Gong, Jun Yu, Gang Niu, and Masashi Sugiyama. Sample selection with uncertainty of losses for learning with noisy labels. *arXiv preprint arXiv:2106.00445*, 2021.

[74] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. In *NeurIPS*, 2020.

[75] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *CVPR*, pages 10687–10698, 2020.

[76] Yi Xu, Lei Shang, Jinxing Ye, Qi Qian, Yu-Feng Li, Baigui Sun, Hao Li, and Rong Jin. Dash: Semi-supervised learning with dynamic thresholding. In *ICML*, pages 11525–11536. PMLR, 2021.

[77] Suqin Yuan, Lei Feng, and Tongliang Liu. Late stopping: Avoiding confidently learning from mislabeled examples. In *ICCV*, pages 16079–16088, 2023.

[78] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

[79] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. In *NeurIPS*, volume 34, pages 18408–18419, 2021.

[80] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2017.

[81] Zhiyuan Zhang, Ruixuan Luo, Qi Su, and Xu Sun. Ga-sam: Gradient-strength based adaptive sharpness-aware minimization for improved generalization. *arXiv preprint arXiv:2210.06895*, 2022.

[82] Yang Zhao, Hao Zhang, and Xiuyuan Hu. Penalizing gradient norm for efficiently improving generalization in deep learning. In *ICML*, 2022.

[83] Yang Zhao, Hao Zhang, and Xiuyuan Hu. Ss-sam: Stochastic scheduled sharpness-aware minimization for efficiently training deep neural networks. *arXiv preprint arXiv:2203.09962*, 2022.

[84] Dengyong Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *NeurIPS*, pages 321–328, 2004.

Supplementary Material for “FlatMatch: Bridging Labeled Data and Unlabeled Data with Cross-Sharpness for Semi-Supervised Learning”

In this Appendix, we provide additional details and experimental results to complement the proposed method. First, we describe supplementary experimental details in Section A. Then, we provide extra quantitative results, including employing FlatMatch to other SSL methods, comparisons on ImageNet30 and ImageNet [55] datasets, performance on Vision Transformer in Section B. Further, we show more empirical results to qualitatively validate FlatMatch in Section C. Then, we conduct convergence study on test accuracy and training loss curves in Section D. Moreover, we provide extra visualizations to show the loss landscape on later state of training in Section E. Finally, we summarize this paper and make a discussion on prospective research in Section F.

A Supplementary Details

The experimental setting of this paper follows Wang et al. [65]. Specifically, the hyper-parameters are composed of algorithm-dependent parameters and algorithm-independent parameters, which are shown in Table 3 and Table 4, respectively. For algorithm-dependent parameters of FlatMatch, we use the same unlabeled data and labeled data ratio as FreeMatch [66] as well as all other baseline methods to sample data into a mini-batch. The perturbation magnitude α is based on the results from hyper-parameter sensitivity analysis in the main paper and is chosen as 0.05 for all experiments. For updating the historical gradient using a memory buffer, we use EMA with factor α to ensemble the gradient result. Moreover, we choose the thresholding strategy from FreeMatch and use an EMA decay. Note that for combining the cross-sharpness regularization from FlatMatch with empirical risk, we find that there is no need to introduce another weight to trade off the two loss functions, hence the weight for cross-sharpness is just set to 1 for all experiments. For algorithm-independent hyper-parameters, we have listed the important model setting, optimizer parameters, and data sampling setting as below. Note that all baseline methods follow the implementation of USB [65] and are trained with EMA decay with 0.999 to smooth the parameter updating.

Table 3: Algorithm-dependent hyper-parameters.

Algorithm	FlatMatch
Unlabeled Data to Labeled Data Ratio (CIFAR-10/100, STL-10, SVHN)	7
Unlabeled Data to Labeled Data Ratio (ImageNet30)	1
Perturbation magnitude ρ for all experiments	0.05
EMA factor α for updating gradient	0.999
Thresholding EMA decay for all experiments	0.999
Trade-off weight $\lambda_{\text{X-sharp}}$ for cross-sharpness	1

B Additional Quantitative Results

In this section, we conduct additional experiments on CIFAR10 and ImageNet30 [55] datasets to compare the performance between some of the most edge-cutting methods, including FixMatch [57], Dash [76], FlexMatch [79], FreeMatch [66], SoftMatch [16], and our FlatMatch.

Table 4: Algorithm-independent hyper-parameters.

Dataset	CIFAR-10	CIFAR-100	STL-10	SVHN	ImageNet30
Model	WRN-28-2	WRN-28-8	WRN-37-2	WRN-28-2	ResNet-50
Weight decay	5e-4	1e-3	5e-4	5e-4	3e-4
Batch size	64				128
Learning rate	0.03				
SGD momentum	0.9				
EMA decay	0.999				

Table 5: Performance on boosting other SSL methods using FlatMatch with the fixed number of labels.

Dataset	CIFAR10		
# label	40	250	4000
FixMatch	7.47±0.28	4.86±0.05	4.21±0.08
FixMatch+FlatMatch	**6.50**±1.25	**4.27**±2.15	**3.92**±1.65
Dash	8.93±3.11	5.16±0.23	4.36±0.11
Dash+FlatMatch	**6.73**±2.49	**4.48**±1.56	**4.02**±1.30
FlexMatch	4.97±0.06	4.98±0.09	4.19±0.01
FlexMatch+FlatMatch	**4.47**±0.92	**4.25**±1.37	**3.88**±0.75
SoftMatch	4.91±0.12	4.82±0.09	4.04±0.02
SoftMatch+FlatMatch	**4.89**±1.32	**3.98**±1.14	**3.84**±0.86
FreeMatch	4.90±0.04	4.88±0.18	4.10±0.02
FlatMatch (Fix label)	**4.89**±1.24	**3.90**±1.72	**3.61**±0.49

B.1 Combining FlatMatch with Other Methods on CIFAR10

We choose CIFAR10 dataset with the number of labeled data varied as 40, 250, and 4000, and apply the FlatMatch methodology to several recently proposed SSL methods to show the effectiveness of the proposed cross-sharpness regularization. The results are shown in Table 5, as we can see that our method can further boost the learning performance of all five methods on all three settings, which proves that the cross-sharpness method is quite universal to SSL approaches and can bring non-trivial performance enhancement. Note that in the 40 labels setting, we compute our cross-sharpness on 500 examples with fixed labels, as demonstrated in Section 5.2 from the main paper.

B.2 Comparing FlatMatch to Other Methods on Large-Scale Datasets and Sophisticated Architecture

To further testify the performance of FlatMatch on a large-scale datasets, we first conduct experiments on ImageNet30 dataset which is a subset from the original ImageNet dataset and contains 30000 training examples with resolution 256×256 from 30 classes. Moreover, we also test the performance on the original ImageNet dataset. As Vision Transformer (ViT) [21] has manifested great power on classification tasks, we also adopt ViT as our backbone to validate the performance of FlatMatch.

The experiments on ImageNet30 are more time-consuming which normally takes 5 days to finish, much more than CIFAR10 dataset which takes 2 days. We vary the number of labeled data as 1500 and 3000 and show the comparison in Table 6. We observe the effectiveness of FlatMatch over all other baseline methods in both two settings, which again validates the superiority of our method and its effective performance on large-scale datasets.

Additionally, we have conducted the experiments on the original ImageNet dataset by choosing only 100 labels per each class, and provide the test error results of FlatMatch, FlatMatch-e, FixMatch,

Table 6: Comparison on ImageNet30.

Dataset	ImageNet30	
# label	1500	3000
FixMatch	12.48±0.67	8.25±0.54
Dash	13.29±1.26	8.79±0.42
FlexMatch	11.48±0.52	8.04±0.75
SoftMatch	10.81±0.40	7.78±0.61
FreeMatch	10.34±0.46	7.21±0.19
FlatMatch	**9.71**±1.55	**6.77**±1.27

Table 7: Comparison on ImageNet using Wide-ResNet-28-2 and Vision Transformer.

Dataset	Architecture	FlatMatch	FlatMatch-e	FreeMatch	FlexMatch	FixMatch
ImageNet	Wide-ResNet-28-2	38.70	39.92	40.57	41.95	43.66
	ViT-Base (86M)	21.57	22.07	23.55	23.78	25.52

FlexMatch, and FreeMatch as shown in Table 7. We can see that on large-scale dataset such as ImageNet, the performances of both FlatMatch and FlatMatch-e are still superior to other components. Moreover, we can still observe the effectiveness of the two of our methods on ViT. Therefore, it is reasonable to conclude that the performance of FlatMatch is extendable to large-scale datasets and sophisticated architectures.

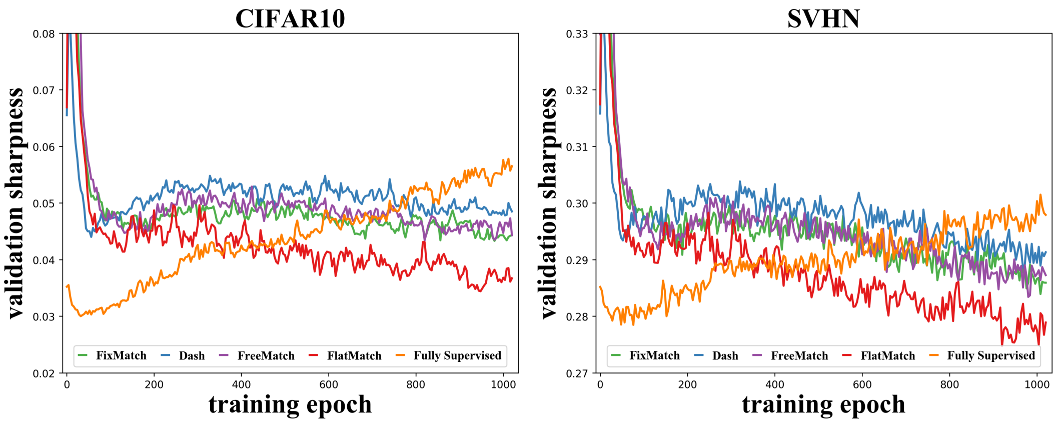

Figure 8: Comparison of sharpness between various SSL methods during training.

C Additional Qualitative Results

To further evaluate the flatness of different SSL models during training, we leverage a validation set to compute the sharpness. The sharpness is measured by the increase of loss within a ℓ_2 bounded neighbor, which is formally defined as $Sharpness := \mathcal{L}(\theta+\epsilon^*(\theta)) - \mathcal{L}(\theta)$, where $\epsilon^*(\theta) = \arg\max_{\|\epsilon\|_2 \leq \rho} \mathcal{L}(\theta+\epsilon)$. Specifically, we compare the proposed FlatMatch with FixMatch, Dash, and FreeMatch, and use fully supervised learning as a baseline method. The experiments are conducted on CIFAR10 and SVHN datasets whose results are shown in Figure 8. First, we observe that FlatMatch achieves the lowest sharpness curve during training on both two datasets, which indicates the SSL model learned by FlatMatch is more robust to perturbations and would not oscillate significantly when facing changes in parameter space. Moreover, we find that fully supervised learning does not improve the flatness as the training proceeds, while all SSL methods can decrease the sharpness to some extent, which demonstrates that training with unlabeled data can help improve the flatness of SSL models.

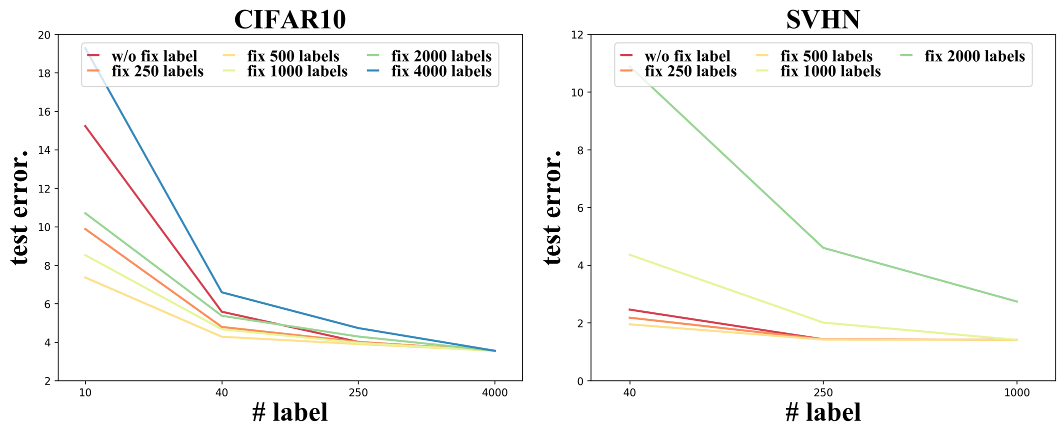

Figure 9: Analysis on changing the number of fixed labels.

Furthermore, as shown in the main paper, we find that FlatMatch has limited performance on extremely scare labeled settings. However, this limitation can be addressed by introducing some unlabeled data with fixed labels to improve the computation of cross-sharpness. Hence, here we investigate the effect of changing the number of fixed on the performance of FlatMatch. Specifically, we conduct experiments on CIFAR10 and SVHN datasets and fixing different numbers of labels as 0 ("w/o fix label"), 250, 500, 1000, 2000, 4000[4]. The results are shown in Figure 9. We find that both too few fixed labels, *i.e.*, 250 labels and too many fixed labels, *i.e.*, 4000 labels in CIFAR10 and 2000 labels in SVHN, would show a performance drop compared to the optimal number, 500 fixed labels. This is because if the number of fixed labels is too small, the gradient computation would be inaccurate, further limiting the learning results. On the other hand, too many fixed labels would introduce noisy labeled unlabeled data, which would largely mislead the SSL and show serious performance degradation.

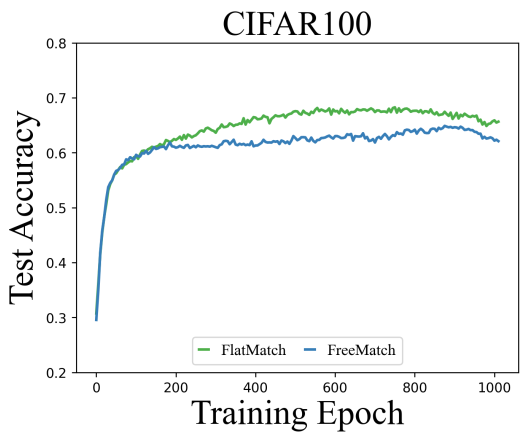

Figure 10: Convergence of accuracy analysis between FlatMatch and FreeMatch.

D Convergence Analysis

In this section, we conduct analyses regarding the test accuracy and training loss to validate the convergence property of FlatMatch.

For convergence of accuracy, here we compare our FlatMatch with FreeMatch, which has the best performance among most SSL baseline methods. The accuracy curve is shown in Figure 10. We

[4]The 4000 fixed labels setting is not conducted on SVHN as the performance of 2000 fixed labels setting already shows significantly performance degradation.

observe that the performances of two methods are almost comparable in early stages, but FlatMatch continues to improve the training performance in the middle and later stages and finally achieves better accuracy than FreeMatch on the final point. Therefore, we can conclude that our method can converge to a superior performance than FreeMatch.

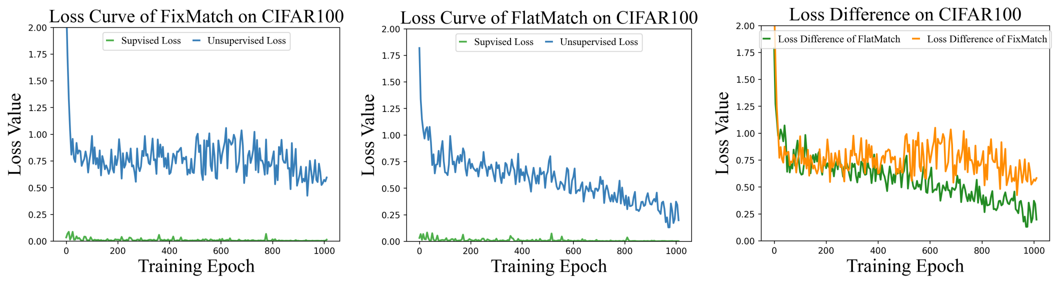

Figure 11: Convergence of loss values of FlatMatch and FixMatch.

Moreover, to illustrate the convergence of loss curves, we show the loss values of labeled data and unlabeled data from both FlatMatch and FixMatch during training. Moreover, we subtract the loss value of unlabeled data with loss of labeled data to compute a loss difference, which gives an illustration about the performance gap between both two datasets.

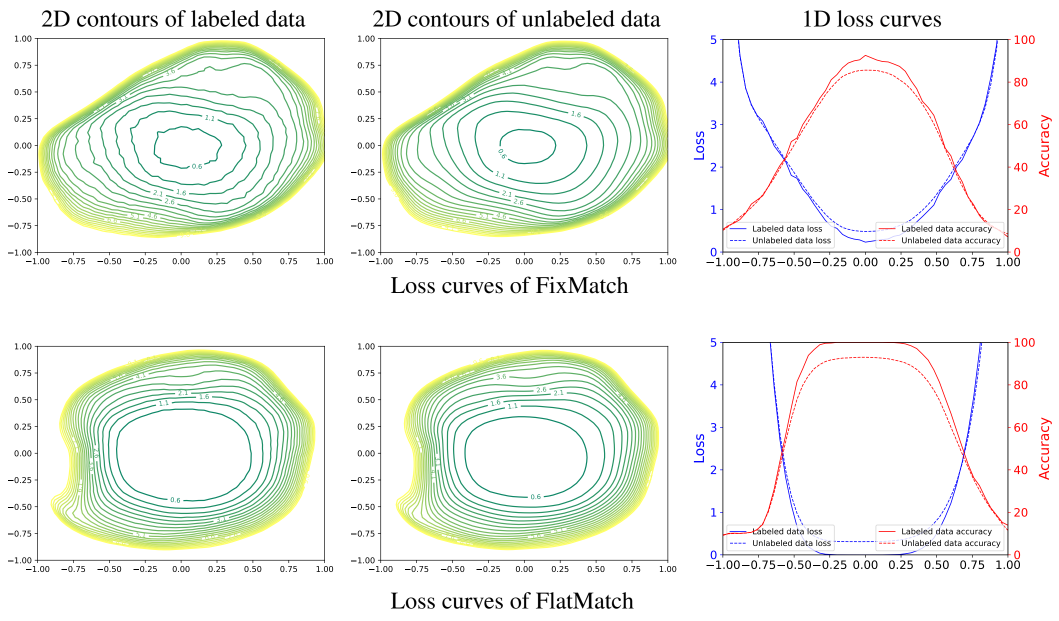

Figure 12: Loss landscapes of labeled data and unlabeled data obtained simultaneously from training using FlatMatch and FixMatch on CIFAR10 with 250 labels per class. The results are generated from the last model checkpoint. The first column and second column show the 2D loss contours of labeled data and unlabeled data, respectively, and the last column shows the 1D loss curves.

There are two findings: 1) The loss value of labeled data quickly converges to zero and is significantly smaller than unlabeled data. Such phenomenon occurs in two methods which supports our claim that the learning on labeled data converges faster than unlabeled data. 2) The loss difference between two datasets of FlatMatch is significantly smaller than FixMatch, which indicates that our FlatMatch can alleviate the unmatched convergence speed of two datasets and helps decrease the loss gap between two datasets.

E More Visualizations

To show how the loss curves appears in the later stage of training, we generate the loss landscape of FixMatch and FlatMatch on the last training epoch (2^{20}). We can see that FlatMatch generates a wider loss landscape than FixMatch. Moreover, the loss curve of labeled data from FlatMatch is smoother than that of FixMatch. Therefore, we can again conclude that FlatMatch can benefit the generalization result.

F Summary and Future Work

In this paper, we propose a novel FlatMatch approach that minimizes the cross-sharpness measure to improve the generalization performance of SSL. Through extensive quantitative and qualitative experiments, we have thoroughly evaluated the performance of FlatMatch and demonstrated its superiority to other compared methods. Thanks to the generalization improvement of FlatMatch, the classification accuracy on many scenarios have even passed the fully-supervised baseline.

However, the learning performance of SSL still largely depends on the careful selection of labeled data. Specifically, in the barely-supervised learning scenario, if the selected scarce labeled data deviate from the cluster center, the learning performance of many existing SSL methods would be significantly affected. This is due to the generalization performance between labeled data and unlabeled data being largely mismatched. Under this scenario, the performance of FlatMatch should be further evaluated.